Name: Anushka Harshavadan Nevgi
Experiment: 8. Implementation annotated tree for example A=B+C B=5 C=5
Class: TY CSE A

```
#include <iostream>
#include <memory>
#include <string>
using namespace std;
// Abstract base class for AST nodes
class ASTNode {
public:
    virtual void print(int level = 0) = 0; // Pure virtual function to print tree structure
    virtual ~ASTNode() = default;
};
// Class for representing the assignment operation (A = B + C)
class AssignmentNode : public ASTNode {
public:
    string variable; // which value is assigned
    unique_ptr<ASTNode> expression; // The expression (B + C)
    AssignmentNode(string var, unique_ptr<ASTNode> expr)
        : variable(var), expression(std::move(expr)) {}
    void print(int level = 0) override {
        cout << string(level, ' ') << "Assignment: " << variable << " =\n";
        expression->print(level + 2);
    }
};

// Class for representing a binary expression (B + C)
class BinaryExpressionNode : public ASTNode {
public:
    string leftOperand;  // Left operand (B)
    string rightOperand; // Right operand (C)
    string op;           // Operator (+)
    BinaryExpressionNode(string left, string right, string op)
        : leftOperand(left), rightOperand(right), op(op) {}
    void print(int level = 0) override {
        cout << string(level, ' ') << "Binary Expression: (" << leftOperand << " " << op << " " << rightOperand << ")\n";
    }
};
// Class for representing variable assignment (e.g., B = 5)
class VariableAssignmentNode : public ASTNode {
public:
    string variable; // Variable name (B or C)
    int value;       // Assigned value (5)
    VariableAssignmentNode(string var, int val) : variable(var), value(val) {}

    void print(int level = 0) override {
        cout << string(level, ' ') << "Variable Assignment: " << variable << " = " << value << "\n";
    }
};

// Function to simulate the entire AST creation and print it
void createAndPrintAST() {
    // Representing A = B + C
    auto expr = make_unique<BinaryExpressionNode>("B", "C", "+");
    auto assignA = make_unique<AssignmentNode>("A", std::move(expr));

    // Variable assignments B = 5 and C = 5
    auto assignB = make_unique<VariableAssignmentNode>("B", 5);
    auto assignC = make_unique<VariableAssignmentNode>("C", 5);

    // Print the AST (Annotated Tree)
    cout << "Annotated AST for the given code:\n\n";
    assignB->print();
    assignC->print();
    cout << "\n";
    assignA->print();
}
int main() {
    // Call function to simulate and print the AST
    createAndPrintAST();
    return 0;
}
```

```
Annotated AST for the given code:

Variable Assignment: B = 5
Variable Assignment: C = 5

Assignment: A =
  Binary Expression: (B + C)
```